Achieving

# write once, run anywhere

in mobile development.

i-CONCEPT

# Table of content

# Introduction

The mobile industry is one of the fastest growing industries. In 2009 more than 1 billion handsets (PDA's, smart phones and such) were shipped[1]. That is four times the amount of personal computers shipped in the same year. People increasingly use the mobile phone to stay connected to the internet, instead of hopping in front of a desktop computer to do so. And why not? We already have it with us all the time. Some people notice on their way to work that they have forgotten their phone and immediately turn around and go home to get it, because it is *that* important. Sounds familiar?

To accommodate the wishes of smart phone users there are online stores that allow them to download applications for their devices. The most popular and successful among these stores is the iTunes App Store from Apple. This store targets the iPhone and iPod Touch devices and exceeds over 100 million downloads every month[2].

The Bright 25 of 2009[3], a top 25 of most important trends, gadgets and innovations of the year 2009, shows us that the number one position is held by 'Apps'. The following is translated from that same source:

> *'The most important development of the last year is the breakthrough of apps. The small applications for smart phones have not only blown new life into the software sector, they have given mobile phones a whole new dimension. The rise of apps slowly started in 2008 when Apple introduced the iPhone 3G and opened the App Store, an expansion of the iTunes Store. Along the way the amount of available applications has grown to 100 thousand.'*

A lot of companies are responding to this still growing trend by starting to develop their own applications. Either to give their existing customers a useful tool for free or simply to sell a very addictive game.

Developing mobile applications is not that easy however. Specifically the fact that all operating systems of smart phones work differently hinders the developers in bringing an application to all smart phones. For example, in order for an application to work on both the iPhone and a Windows Mobile device the developer has to write the application twice, because the two operating systems use different programming languages.

[1] Global Handset Shipments Fall 4 Percent in Q3 2009, but Return to Growth Expected in Q4
[2] App store downloads exceed 100 million per month
[3] Bright 25 of 2009

## Construction of whitepaper

The found problem "write once, run anywhere" in developing mobile applications has led to the following main research question that will be discussed in this whitepaper:

**How can a company not already familiar with mobile development achieve 'write once, run anywhere' to create mobile standalone applications for smart phones?**

In order to answer the main research question, a few sub-questions that are relevant to the problem have been formulated and will be discussed in several chapters. On the next page these chapters will be summarized.

**Chapter1:** Smart phones.
*"What is a smart phone?"*

Since there is no solid definition of a smart phone a review of the properties of the target devices is shown below, hence creating an own definition of a smart phone that will be used throughout this whitepaper.

Chapter 1 – Topics:

1. Operating System (OS).
2. Internet access.
3. Touch screen.
4. QWERTY keyboard.
5. Software.

**Chapter 2:** Native applications versus web applications.
*"Why choose for standalone applications instead of web applications?"*

This chapter will discuss the differences between standalone (sometimes also called native) applications and web applications. Some popular applications will be highlighted to show why developing standalone (or native) applications can be interesting.

Chapter 2 – Topics:

1. What is a standalone application?
2. What is a native application?
3. What is a web application?
4. What are similarities and differences between the three?
5. What are popular standalone applications and why are they popular?
6. What is the use of developing standalone applications?

## Chapter 3: Current methods of developing mobile applications.

*"How are mobile applications currently being developed?"*

This should give some insight to what programs and methods are being used to write applications for different smart phones. For example, some companies only develop for the iPhone, because it is the most successful and will thus make the most profit. Mainly the technical possibilities and limitations will be discussed. Related information like a target group for an application will be mentioned, but not in full detail.

Chapter 3 – Topics:

1. Developing for mobile devices.
2. What developing platforms are available?
3. The native languages of the five major operating systems.
4. Can the goal be achieved with the current methods of development?
5. Do these current methods of development have a future?

## Chapter 4: New ways of developing mobile applications.

*"What are upcoming technologies that allow to create mobile applications more easily in the future?"*

Which technologies are being developed (or have been released recently) and what are their opportunities? A lot of companies are already trying to find (and may even have) a solution to the problem. Some companies are well-established, others are brand new. How do they plan to tackle the "write once, run anywhere" problem? Quite a few projects and/or services will be discussed in this chapter, mentioning strengths, weaknesses and more to distinguish one from the other.

Chapter 4 – Topics:

1. What upcoming methods might be available in the near future?
2. Could the goal be achieved with any of these platforms?

After these four chapters, it should be clear what is possible and what is not. Let's start a case study and put the research into practice.

**Case study**

With all the information gathered, it's possible to start building an application. In this case an application in which it will be possible to read a news feed on the user's phone and which allows to submit news as well.

**Conclusion**

The case study will show what was helpful and what was not. Based on the results of the research and the case study some conclusions can be made on developing methods. Can an answer to the question be discovered or will it remain unanswered for the time being?

# Chapter 1:
# Smart phones.

*"What is a smart phone?"*

There are hundreds of different phones people can choose from. Brands like LG, Samsung, Nokia and Sony Ericsson (and many more) each have several new models coming out each year. What makes a mobile phone a smart phone?

There is no standard definition of a smart phone, so some of the features that will explain what makes a smart phone different from others are listed below. Not every smart phone out there will have these features, but with these features we describe the target group of mobile devices.

## 1.1 – Operating System (OS).

Unlike the older phones that run an operating system from the device manufacturer itself, smart phones can run on one of these operating systems:

- Windows Mobile – developer: Microsoft.
- Android – developer: Open Handset Alliance.
- OSX – developer: Apple.
- Blackberry OS – developer: Research In Motion (RIM).
- Symbian OS – developer: Symbian Foundation.

There are more operating systems available, but these are the five major ones and cover the majority of smart phones. The ones with these five operating systems are the smart phones targeted in this whitepaper.

## 1.2 – Internet access.

By our own definition, smart phones have to be able to connect to the internet. This gives the user the ability to browse the web and allows applications to send and retrieve data from outside sources. Not every smart phone has a high-speed connection, but all of them at least offer some access.

## 1.3 – Touch screen.

Smart phones have a touch screen. A touch screen is a screen that detects where and when the user touches it with his/her finger or a passive object like a pen. Users will not need to use physical navigation keys, but can do (almost) everything by touching the screen.

## 1.4 – QWERTY keyboard.

Smart phones have a QWERTY keyboard. This means that the keys have the same layout as a computer keyboard, not in alphabetical order on the number keys. The QWERTY keyboard may either be physical keys (hardware) or used with the touch screen (software on screen). The pictures shown below are an iPhone with a touch screen keyboard (left) and an HTC Touch Pro with a physical keyboard. Both are QWERTY.

## 1.5 – Software.

Almost all cell phones (even the older models) have a way to manage their contacts. This is a piece of software usually pre-installed on the device. With smart phones, you have the option of installing your own software. Either by writing applications yourself or downloading them from an online store. The fact that you can "personalize" your smart phone is probably the most important feature.

So, by our own definition, a smart phone runs on one of the five major operating systems, has access to the internet, has a touch screen, a QWERTY keyboard and allows the user to install new software (apps).

Now that our own definition of a smart phone has been established, let's explore the area of applications. Personalizing your smart phone with apps is what contributes to the immense success of the iPhone. The next chapter is dedicated to applications.

# Chapter 2:
# Standalone applications versus web applications.

*"Why choose for standalone applications instead of web applications?"*

Please note that only applications for mobile devices such as smart phones are being discussed. At the end of this chapter people should have an idea about why the focus lies on standalone applications. The popular applications section is there to inform about the opportunities that applications can offer to companies that are interested in mobile development.

## 2.1 – What is a standalone application?

A standalone application is a program that can be installed onto your mobile device. After installation the program will be listed among all other programs that exist on your mobile device. When you run the application it will start in a new screen. It does not require a web browser to run in (unlike the web application).

## 2.2 – What is a native application?

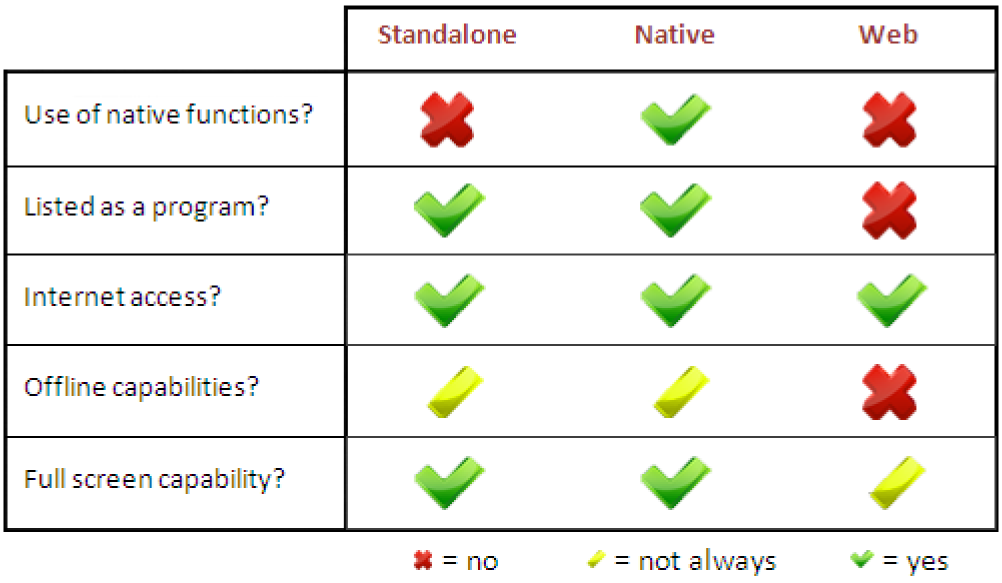
A native application has all the properties of a standalone application, but is allowed more: it can use the native functions of the mobile device, like the camera, global positioning system (GPS), short messaging service (SMS), the phone function and more. Because a native application includes the properties of a standalone application, it also will appear in the list of installed programs on the device and will run in its own screen (usually full screen).

## 2.3 – What is a web application?

A web application does not have to be installed, but can only be accessed whenever the user is connected to the internet. Like the standalone application, a web application does not have access to the device functions like the camera and GPS. One major advantage is that a web application runs in a browser, meaning that it most likely will run on any smart phone.

## 2.4 – What are the differences between all types?

The following comparison table should give a good picture of the differences between the three types of applications.



| | Standalone | Native | Web |
|---|---|---|---|
| Use of native functions? | ✗ | ✓ | ✗ |
| Listed as a program? | ✓ | ✓ | ✗ |
| Internet access? | ✓ | ✓ | ✓ |
| Offline capabilities? | ◐ | ◐ | ✗ |
| Full screen capability? | ✓ | ✓ | ◐ |

✗ = no ◐ = not always ✓ = yes

## 2.5 – What are popular applications and why are they popular?

Applications can be divided into different categories. Not just by genre (games, business, social), but also by cost (€0,99 or €5,99). The majority of all applications that are downloaded are games, but that doesn't mean that the best applications are all games. Does popularity mean a good review rating or are the applications ranked by total amount of downloads? If you rank them by downloads, the cheap applications will usually be more "popular" than the more expensive ones. Let's look at applications for the most successful operating system, who started the "application trend". We do this because the iPhone has far more applications available in its App Store than Windows Mobile or Android. All the applications mentioned in this topic are found on the Apple iTunes App Store.

### iPhone

To find and download applications for the iPhone, one has to use iTunes. It is hard to find the popular applications though, because they are generally ranked by number of downloads (though not always). The number one "popular" free application is a flash light, with a 1-star rating (out of 5 stars). More than 70% of the users who downloaded the application gave the application the lowest rating possible. Even though these applications get low ratings, the initial idea of the application is usually what appeals to the users who download them. Low ratings often are the result of bad user experience (like bugs or incomplete information). Nonetheless the 'most popular' applications from iTunes will be looked at in order to try to determine what users are looking for in an application.

One of the most popular applications at the moment is Ping! It allows you to send instant messages to other iPhone users who have Ping! and it will cost you nothing. Even if you are on the other side of the world you can contact each other without cost. With this application people can stay in contact with their social network (at least with those who have iPhones too). A possible reason for its popularity is the fact that it is free to send messages, unlike SMS that charges per message sent (and sometimes even per message received).

Mobile games like The Settlers, Fling! and Doodle Jump are also in the top 10 paid applications. The first is a game for PC that has been reworked for the iPhone. Despite its relative high cost (€3,99 compared to the usual €0,79) it is the best selling application in the iPhone App Store. The other two are fun games that are highly addictive. In the top 30 of paid apps, 10 are games. So game apps sell well, but all depends on how fun, cheap, complicated or addictive they are.

Another interesting category is "service". From the same top 30 paid apps, seven applications (including Ping!) fall under this category. An interesting mention is "Belstatus". This application allows T-Mobile users to track their calls and the text messages they send. It also shows the remaining minutes and text message and will warn you when you are close to using up your minutes. This application is a convenience for T-Mobile customers. People who download this are probably interested in their mobile phone behavior and/or really want to have more detailed information of their T-Mobile bundle usage.

Another popular application is P-2000. This acts like a standard newsreader, but it only shows alerts from ambulances, the fire department and the police department in specific regions. People who download this application may simply be curious about what's happening in their region or it could be a hobby.

Air Mouse Pro is an application that (in combination with its PC server software) lets you use your iPhone as an intelligent wireless mouse. If you hooked your PC to your TV, you can use your iPhone to navigate effortlessly without having to leave your seat.

One of the highest rated applications is "Trein" (Train). This application  uses the internet and gives you all the information you need when travelling by train. It tells you whether or not your train will arrive on time, whether or not you need to catch another connecting train or if you forgot to get out (in combination with another application: iNap, that will generate an alert) based on your GPS location and the choice of destination.

### Summary

Summing up a few possible factors that might be why applications are successful.

- An application is *remarkable* and does something *unique* that other applications can't do.
  Example: Shazam recognizes millions of songs by just "listening to" music a few seconds.
- An application is *extremely fun* and *addictive*.
  Example: Fling! uses the touch screen in order for the user to solve complicated puzzles .
- An application provides an *existing (computer) service* that people are looking forward to using on their *mobile* device.
  Examples: Facebook and Skype are both very popular and already widely used on mobile devices.
- An application is *a convenience* to the user.
  Examples: Check your train schedules with Trein or control your PC from a distance with your iPhone with Air Mouse Pro.

Now we know that applications are hot and have seen some examples of popular apps. These are all end-products though. How did they end up in the app store, how were they developed and deployed? The next chapters will go through several methods of development and in particular compare them, in order to discover which ones are best suited to start developing mobile applications with.

## 2.6 – Why develop standalone applications?

As shown in the comparison table above, native and standalone applications are more versatile than web applications. Web applications need internet access in order to work. Not all web applications need internet access the entire time they run, but still an active internet connection would be preferred.

The only real noticeable difference between standalone and native applications is that standalone applications can't make use of the native functions of the phone. This might be a reason to choose to develop native applications, but not every developer wants to use the native functions of the device for his/her application.

Assumed is that there are more development platforms and/or technologies that allow to develop standalone applications for smart phones than there are for writing native applications. This is one of the reasons for choosing standalone applications.

Another reason is that standalone applications do not require an active internet connection (perhaps once to download and install). Mobile games can usually be played while not connected to the internet, although they often do have an option to connect to the internet (to submit high scores for example). Not requiring an internet connection, but having the option is a big plus.

# Chapter 3: Current methods of developing mobile applications.

*"How are mobile applications currently being developed?"*

As this whitepaper is targeting companies that are not already familiar with mobile development, assumed is that these companies also are not familiar with the development tools that are addressed in this chapter or the limitations that mobile devices have. The good news is that there are plenty of options, the bad news is finding one that suits the needs. So let's get started.

## 3.1 – Developing for mobile devices.

Compared to a computer monitor the screen of a mobile devices is very small. The HTC HD2 for example has a 3.4" screen, which is already relatively large compared to most other smart phones. A small screen means smaller visual elements, so one should not forget to take that into account.

Most smart phones nowadays have a touch screen. Again, the size of elements on the screen shouldn't be too big, but they shouldn't be too small either! One has to be able to 'click' a button with his/her finger, without hitting something else instead. It can be compared with the small physical keys on a mobile phone. People with big hands (and fingers) have a hard time punching the tiny keys. Try and prevent users from this experience.

## 3.2 – What developing platforms are available?

Several platforms will be discussed, pointing out strengths and weaknesses when listing their features. Some platforms are well known and widely used (like Java ME and Flash Lite), others are less popular, but might prove to be very useful. At the end of this topic a review can be made about each platform by checking out the comparison table.

### 3.2.1 – Java ME (Sun Microsystems)

Java ME (Micro Edition) is a Java platform designed by Sun Microsystems for mobile devices and embedded systems. It was formerly known as J2ME. Mugdha Chauhan, a mobile games developer, listed the following about J2ME in an online article[4] in 2005:

- J2ME is a free and open platform. This helps keep the development costs low and provides the necessary flexibility with ample support freely available for developers using it.
- J2ME enjoys the status of an industry standard backed by all major handset makers, with most of the present day mobile phones being Java-enabled.
- Its highly portable nature ("Write once run anywhere") ensures that a game application written for one brand/type of handset will work with all other brands/types of Java-enabled handsets.
- It is especially optimized for small devices, is lightweight, and is highly secure because applications written on it cannot access or affect other applications running on the phone/device.

Java ME has been very popular because it was widely adopted by the major manufacturers. Today the majority of all phones are still Java enabled. Chauhan mentioned "write once, run anywhere", which Sun Microsystems might have achieved in 2005. Today however there are a lot of devices that are not Java enabled. Windows Mobile does not support Java. The iPhone does not run (and cannot install) Java applications. Android doesn't recognize them without a proper emulator. Only devices with a Java Virtual Machine (basically the 'software' to run Java applications).

So even though Java ME might have been extremely popular once, it is now less attractive to mobile developers due to it losing terrain in the smart phone market.

Nonetheless, it is still a very viable platform to build applications and reach millions of phones. A lot of developers still choose to use this platform. Especially mobile games that target non-smart phones are often Java ME applications.

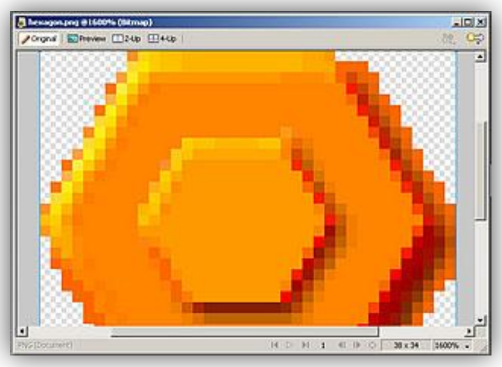
[4] Developing Java-Based Mobile Games

### 3.2.2 – Flash Lite (Adobe)

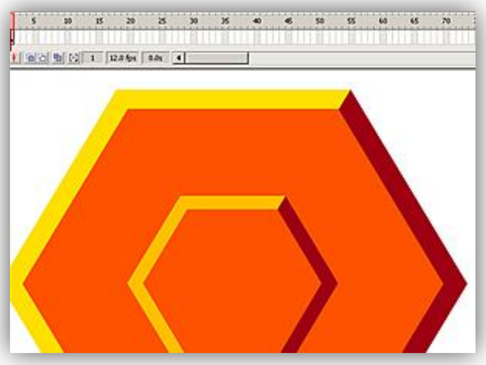
Flash Lite is the little sister of Flash, specifically designed to run on mobile devices. With the help of Flash Lite one can create applications for devices that support it. Before Flash Lite was introduced most developers made use of J2ME (see previous topic: Java ME), because it was supported on mobile devices worldwide by almost all mobile device manufacturers. In one of his articles on ZDNet[5], Ed Burnette (a professional developer) tells us:

*"Flash Lite is extending the Flash ecosystem to mobile devices. What was previously the domain of J2ME (now called Java Micro Edition or Java ME) is now being Flash-enabled. Because Flash is vector-based, applications are easier to port from one screen size to another, and users don't see jaggies because Flash has always done anti-aliasing."*

One of the more important things that makes Flash Lite so interesting is mentioned here: Flash is vector-based. Vector-based graphics make use of geometrical primitives which are all based on mathematical equations, to represent images[6]. The advantage of vector-based graphics are best shown with an example. The following illustrations should help you understand the difference between standard images and vector-based ones.



Normal image (pixels)



Vector-based image

The image on the left is 'jaggy'. Edges aren't smooth and every pixel is clearly visible. When zooming out it will be less noticeable and it will seem more smooth, but when zooming in the pixels will become more clearly. The image on the right however will always have smooth edges, no matter how far zoomed in or out, because the image is vector-based. Applications created in Flash can work on different screen sizes without losing any image quality.

[5] **Is Flash better than Java?**
[6] **Vector graphics**

The real advantage of Flash is the amount of devices that support it. It is suspected that at the end of 2009 there are over 1.5 billion devices that support Flash Lite worldwide.

Unfortunately, not all mobile devices (and more specifically, smart phones) have the Flash Player runtime. Even though Windows Mobile supports Flash Lite, the iPhone does not. Most Symbian smart phones support Flash Lite, but the Blackberry does not allow installation of applications in Flash Lite format.

### 3.2.3 – Silverlight (Microsoft)

Silverlight is Microsoft's reply to Adobe Flash. Microsoft defines Silverlight as follows[7]:

*"Microsoft Silverlight is a cross-browser, cross-platform, and cross-device plug-in for delivering the next generation of .NET based media experiences and rich interactive applications for the Web. On mobile devices, Silverlight provides a homogenous platform for developers to target a large number of devices as well as deliver rich interactive applications with scalable vector graphics UI and mobile-optimized media."*

Currently Silverlight is in private testing phase, so it has not been released yet. It appears that only Windows Mobile and Symbian S60-series are supported Silverlight when it will be released, which will most likely be early 2010.

### 3.2.4 – QT (Nokia)

The description Nokia gives us of its own service is promising[8]:

*"Using Qt, you can write web-enabled applications once and deploy them across desktop, mobile and embedded operating systems without rewriting the source code."*

That sounds good. Let's take a deeper look at the features of Qt.

With the help of Qt and the Qt Designer one can write and design programs quite easily. Knowledge of the C++ programming language is required though. Qt Designer is a flexible user interface builder that supports IDE integration and is used to create the visual interface of your application.

[7] Silverlight for Mobile
[8] Qt – A cross-platform applications and UI framework

To deploy an application on multiple operating systems, Qt makes a recompilation of the source code using the native programming language of different operating systems. More information about these native programming languages will follow further on in this chapter.

For now, Qt supports recompilation for Windows mobile and Symbian devices only.

### 3.2.5 – Mobile AJAX

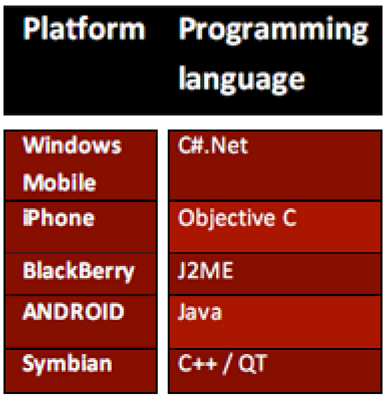
The term 'mobile AJAX' came across more than once while browsing on the internet and provided the following information. AJAX (short for Asynchronous JavaScript And XML) is a very popular choice among web developers and is well known, so one could say it's only logical it would be associated with mobile at some point. However, it is only used for web development. AJAX could definitely reach the majority of the target devices through web applications, but as the focus lies on standalone applications this is not an option.

### 3.2.6 – More alternatives

There are more alternatives out there, but unfortunately there is too little information available to give an idea of what they are capable of. An example is JavaFX, that seems to have some support from developers, but it seems to only be available for a small range of devices and has yet to work on the five major operating systems.

## 3.3 – Native languages of the five major operating systems.

A native programming language is the language that a device can understand without the need of an interpreter, like the Flash Player or Java runtimes. Often a device only allows native code to access specific functions, the camera for example. The five major mobile operating systems all use a different native language, as shown in the image on the right.



| Platform | Programming language |
|---|---|
| Windows Mobile | C#.Net |
| iPhone | Objective C |
| BlackBerry | J2ME |
| ANDROID | Java |
| Symbian | C++ / QT |

Native languages for the five major operating systems.

In short: every operating system has its own native language, which does not make it easier for developers to make applications for all of them. Should they focus on the Java-oriented platforms like BlackBerry and the Android? Or for the iPhone with its Objective C?

The optimal situation would be that developers will not have to choose. They should be able to write the application once and then not have to worry about whether or not it will work on all five major operating systems.

## 3.4 – Can the goal be achieved with the current methods of development?

It can be concluded that ‘write once, run anywhere’ cannot be achieved with the methods discussed in this chapter. All of them have their pros and cons, but all of them share the disadvantage of not being able to run on all five major operating systems.

Flash Lite cannot run on an iPhone, J2ME does not work on a Windows Mobile or Android device without a proper emulator (which could be described as an interpreter), Qute (Qt) is limited to Windows Mobile and Symbian devices. Silverlight is too new and underdeveloped to really start making a difference in the field and even then they still require a runtime environment, just like Flash Lite.

## 3.5 – Do these current methods of development have a future?

Sun Microsystems does not have concrete plans for the Java platform on mobile devices, at least regarding standalone applications. They are trying to encourage developers to try JavaFX, but one can only write mobile web applications with it. It seems that Java ME will very slowly disappear from the world of mobile development.

Adobe has big plans for the Flash platform is continually trying to get the runtime to ship on all major operating systems. Their hope lies in Adobe AIR for mobile (discussed in chapter four) and their new release of the Flash Lite player (Flash Lite 4.1). Both of these should be released in 2010. So Adobe is still very much in the game and should be kept an eye on in the future.

Microsoft is getting more and more support from developers who are infatuated with Silverlight. Once Silverlight for mobile will be available, we can really see how it will position itself in relation to Adobe Flash Lite (or AIR). Any predictions for it cannot be made right now, but Silverlight should not just be overlooked for mobile devices.

Nokia has a good concept with their QT tool. Right now it only supports Windows Mobile and Symbian devices, but undoubtedly they will add support for all three other operating systems as well. This might be the prime candidate to achieve ‘write once, run anywhere’ in the future.

It looks like an answer to the research question cannot be found so far. Let’s research some new and upcoming possibilities and put hope in the future of mobile development.

# Chapter 4:
# New ways of developing mobile applications.

*"What are upcoming technologies that allow to create mobile applications more easily in the future?"*

So how is the goal (write once, run anywhere) achieved when the current possibilities do not fulfill the needs? Are there companies that are trying to achieve the same thing? Perhaps one of them has already thought of a way to do it and is just waiting for the right time to release their product. In this chapter a few platforms that were stumbled upon will be discussed. Most of the platforms are in a testing phase, but the companies claim that their product allows you to create applications for every smart phone. Let's see if they can deliver that promise.

## 4.1 – What upcoming methods might be available in the near future?

Write once, run anywhere. It seems to be the goal of many developers. Adobe has taken point in the quest to make this happen by leading the OpenScreenProject[9], an industry-wide initiative to provide a consistent runtime environment for open web browsing and standalone applications — taking advantage of Adobe Flash Player and, in the future, Adobe AIR. Many companies and smart phone manufacturers have joined the OpenScreenProject, including RIM (Blackberry), Google (Android) and Nokia (Symbian). This initiative shows that there will definitely be a solution to our problem in the near future, possibly even faster than we might think.

Let's take a look at the options we might have in the future.

### 4.2.1 – AIR Mobile (Adobe)

One of the aims of the OpenScreenProject is to bring the AIR runtime to mobile devices. What exactly is AIR? This is what Adobe says[10]:

*"Adobe AIR is a cross-operating system runtime developed by Adobe that allows developers to leverage their existing web development skills (Flash, Flex, HTML, JavaScript, Ajax) to deliver web applications beyond the browser."*

The fact that developers can use their existing knowledge to write cross-platform applications sounds intriguing. Unfortunately, Adobe AIR is not yet available for mobile devices, but it promises to be valuable for developers in the future.

[9] Open Screen Project
[10] Developer FAQ – Adobe AIR

Even though Adobe AIR cannot be tried out for mobile yet, the fact that the OpenScreenProject is backed by a lot of big companies shows that they have a lot of faith in the future of AIR for mobile. Some of the partners in this project are:

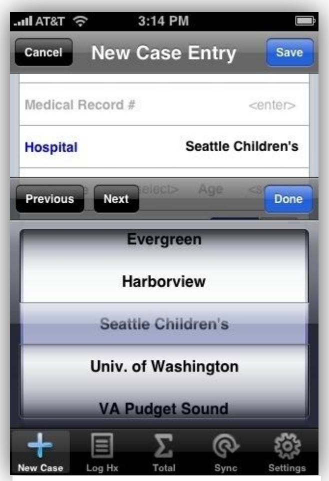
- Nokia
- HTC
- Motorola
- Google
- RIM
- Sony Ericsson
- Palm
- ARM

These are all quite renowned companies and a lot of them are in fact even manufacturers of mobile phones.

### 4.2.2 – Rhomobile

Rhomobile's programming framework (called Rhodes) allows to write applications for mobile devices using only HTML and Ruby. That's right, no need to learn C#, Objective C, Java or C++. Most developers are familiar with HTML and according to Anthony Ha[11] learning Ruby is a lot easier than Objective C or any other native language for smart phones. But will the applications built with Rhodes run on the five major operating systems?

Rhomobile's answer is yes. Once an application is written (in HTML and Ruby) and is ready to be deployed for the iPhone, it will be recompiled into the native language specifically for the Apple device and it will even have customized looks to match. For example, a list element in the application will be shown as the familiar scroll wheel on an iPhone. If the same application is deployed for Windows Mobile, that same list element will look completely different from the iPhone one.



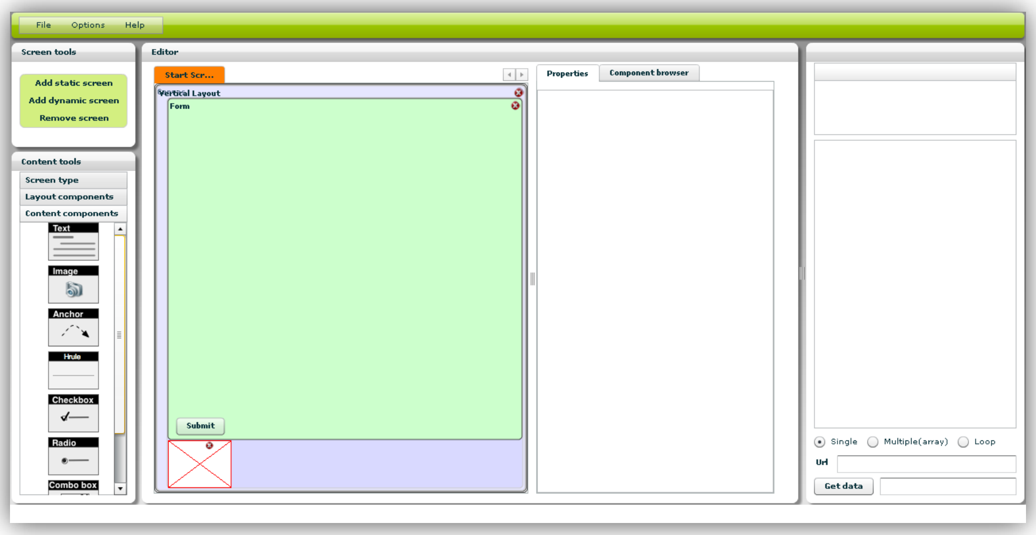
The scroll wheel on iPhone.

If Ruby is indeed easy to learn, then Rhomobile might be very interesting to companies who are interested in developing mobile applications. But there are still other options to evaluate.

[11] Rhomobole promises: Build once, deploy to any smartphone

### 4.2.3 – Steape

Steape takes a similar approach as Rhomobile, but there's a big difference between the two. The difference is the drag and drop functionality of Steapes online development platform. So instead of having to learn Ruby, one only has to understand the user interface of Steapes online platform in order to start writing applications.



A screenshot of the Steape platform user interface (drag & drop usability).

Steape allows to upload own content (like a logo, images and splash screens), so it's also possible to visually edit the applications to your own likings. Just like Rhomobile, certain elements (like lists) will have the proper look for each operating system, without you having to change it.

At the moment Steape has not yet been officially released to the public, but a select few have been allowed to try out the online platform by means of a demo account.

A Dutch company called IENS has already successfully released three versions of their application[12]. They are available for the iPhone, Windows Mobile and Android devices. They used Steape to develop and deploy their application.

[12] De nieuwe IENS applicatie nu ook gratis voor Android- en Windows Mobile-toestellen

### 4.2.4 – Elips Studio 3 (OpenPlug)

Elips Studio 3 is a plug-in for the popular Adobe Flex Builder IDE. Flex Builder works with ActionScript and MXML (which is basically XML that will be converted to ActionScript as well). Both of these languages are relatively easy to learn. Basically what Elips Studio 3 provides is a set of libraries that Flex Builder can use to allow placement of UI components and the use of native phone functions, like sending SMS or contacting the address book.

This is what OpenPlug says about its own product:

> *ELIPS Studio 3 automatically mobilizes and packages your Flex-based applications for industry-leading platforms, including iPhone, Android, Symbian, Windows Mobile, plus proprietary mass-market devices that run Real-Time operating systems (RTOS).*

So Elips Studio 3 again recompiles data into native languages, but it seems it is missing support for the BlackBerry. This method is mentioned nonetheless, because it covers four out of five operating systems and provides working with Flex Builder, which has already proven to provide a good user experience when developing applications.

## 4.2 – Can the goal be achieved with any of these platforms?

Adobe is still working on bringing AIR to mobile devices, so for now that is not an option. Possibly in the future, but there is no guarantee. The other three promise great things. If they can deliver on these promises, two of them could be used to achieve ‘write once, run anywhere’. Elips Studio 3 however would be missing support for the BlackBerry, so that one is left out as well. These two options will be evaluated:

- Steape
- Rhomobile

## Case study

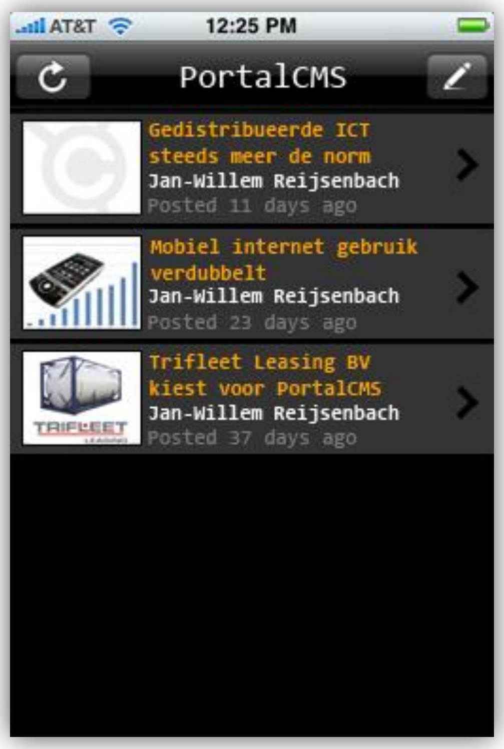
As mentioned earlier, both Rhomobile and Steape are not fully operational yet. Rhomobile launched their services in March of 2009, Steape started handing out demo accounts around October that same year. They still have a lot of improvements to make and bugs to fix. In this case study an application will be written, that will have to work on several smart phones.

### Assignment

The application that will be written is one that shows weblog items and allows to post new items too. Since this is about the way we are developing and not about usability or interaction design, the application isn't going to be too complicated. Only the weblog items of one source will be shown and also will allow new items to be posted to that same weblog. The destination weblog will be static and cannot be changed. There is the possibility to add more features to the application at a later stage, but that will not be described in this whitepaper.

### Developing process

Even though it is about developing methods, interaction design cannot totally be ignored. The first thing to do would be to create a flowchart, but this step is going to be skipped because the application that is going to be written is too simple. The next logical step would be to make wireframes (projections of what the screens will look like). More detailed images have been made in Photoshop to represent how the application will look on a smart phone. The pictures are shown below. The content is taken from news on actual website.

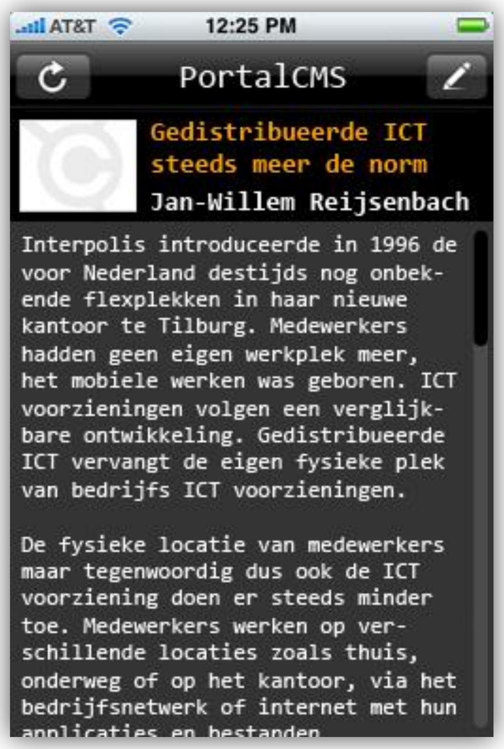


Gedistribueerde ICT steeds meer de norm
Jan-Willem Reijsenbach
Posted 11 days ago

Mobiel internet gebruik verdubbelt
Jan-Willem Reijsenbach
Posted 23 days ago

Trifleet Leasing BV kiest voor PortalCMS
Jan-Willem Reijsenbach
Posted 37 days ago

First screen, shown when application is started.

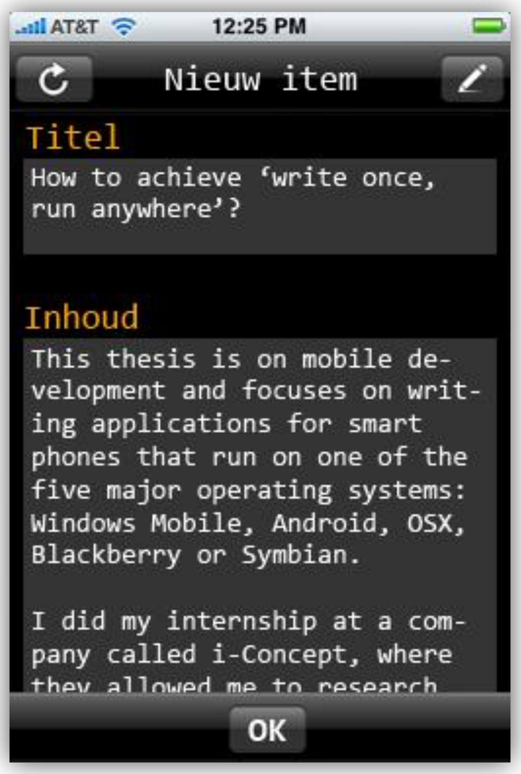


Gedistribueerde ICT steeds meer de norm
Jan-Willem Reijsenbach

Interpolis introduceerde in 1996 de voor Nederland destijds nog onbek-ende flexplekken in haar nieuwe kantoor te Tilburg. Medewerkers hadden geen eigen werkplek meer, het mobiele werken was geboren. ICT voorzieningen volgen een vergelijk-bare ontwikkeling. Gedistribueerde ICT vervangt de eigen fysieke plek van bedrijfs ICT voorzieningen.

De fysieke locatie van medewerkers maar tegenwoordig dus ook de ICT voorziening doen er steeds minder toe. Medewerkers werken op ver-schillende locaties zoals thuis, onderweg of op het kantoor, via het bedrijfsnetwerk of internet met hun applicaties en bestanden

Reading the latest web log entry.



How to achieve 'write once, run anywhere'?

This thesis is on mobile development and focuses on writing applications for smart phones that run on one of the five major operating systems: Windows Mobile, Android, OSX, Blackberry or Symbian.

I did my internship at a company called i-Concept, where they allowed me to research

Writing a new web log entry.

These screens are pretty self explanatory, but let's go through them quickly anyway. The first screen (image on the previous page, on the left) is what people will get to see when they start the application. It lists the entries of the weblog, with the most recent entry always being on top. The two buttons in the top are the 'back' button (left) and the 'write' button (right). The first will go back obviously. In this case it will exit the application. The second will direct to the screen that allows to write a new entry.

The second screen (image on the previous page, on the right) shows an entry which is selected to be viewed. The whole post will be shown then. If the post is too long to fit on the screen, it is possible to scroll down to view the rest of the text. It has the same two top buttons. This time when the back button is pressed, it will return to the first screen. The write button will always act the same and will direct to the write screen.

The last screen (shown on this page) is the write screen. When the title and content of the information that you want to post is filled in, the content can posted by clicking on 'OK'. It will send and store the inserted data (the text) and redirect to the start screen again. The post should immediately be shown at the top, since that is now the most recent entry in the list. The back button will also redirect back to the first screen.

Now a clear (visual) view has been formed on what kind of application needs to be created. Keep in mind that these images might not represent how the application will look like in the end. The goal of course is to get a result that comes as close to this as possible, but it all depends on how Rhomobile and Steape handle the graphical user interface. It might be really simple to get a great result, but it might also be a huge problem. That will have to show once the application is done.

So what is actually needed to write this application?  What components (like a text field) are going to be used, is internet access going to be required? These questions might seem irrelevant, but they might help understand how the application is going to be build in Steape and Rhomobile. Let's sum up what is required on the next page.

From the three screens that were shown on previous pages, a few components can already be named. Obviously there is a lot of text involved, so text fields will definitely be needed. Now there is a difference between a static and a dynamic text field. The first will always have the same value (text) and cannot be changed. The second is dynamic and will load its value from an outside source or from variables inside the application itself. For example: if a button is created and a counter needs to go up each time that button is pressed, a dynamic text field will be needed. Every time the button is pressed, the application will calculate (invisibly and internally) how often it was pressed and knows the value of it. That value will then be displayed in the dynamic text field.

Anchors to navigate between screens and give commands to the application are also needed. An anchor is basically a reference to a specific screen or command. An anchor could be a clickable image (usually called a button) or clickable text. Website links that are seen all around the internet are usually clickable text anchors. Anchors can redirect to another screen or submit a form (and even both at the same time).

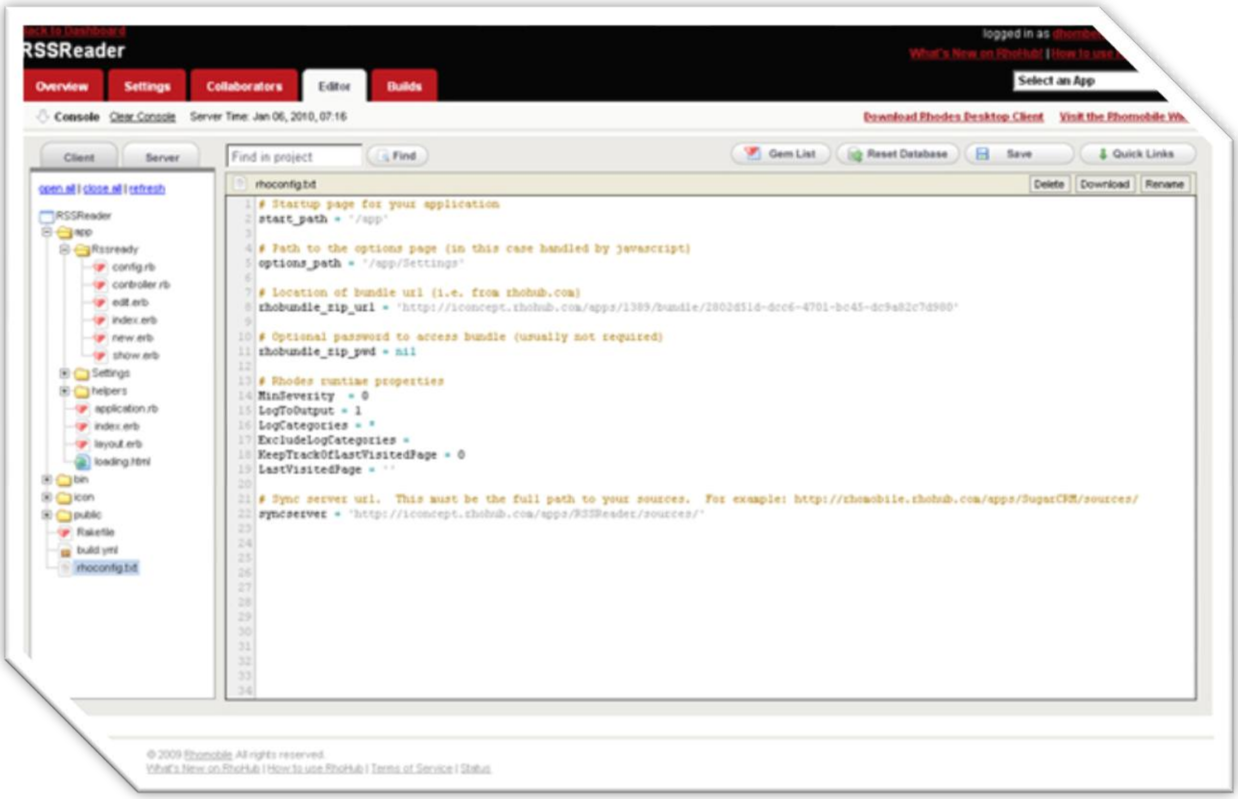
Images were mentioned above. They are either pictures or can be colored rectangles, but when used correctly they usually enhance the graphical user interface. Images that were already showed in the three screens are going to be used. One for the back button and the one for the write button are already visualized. Some images will be loaded dynamically, because they will be different for each web log entry (as is the same for some of the text fields).

The minimal requirements component wise are now checked. Some things that will happen will be invisible (like requesting data from online) and are executed by some code. Rhomobile will handle this differently than Steape, because the first requires to input the code by hand and Steape has more of a drag and drop approach and then lets you make some property adjustments. Let’s start writing the application and start with Rhomobile.

### Rhomobile

Rhomobile works with the programming language called Ruby. It is not a familiar language, so let’s get an idea how it is different from other languages. In chapter 4.2.2 it was mentioned that a developer named Anthony Ha claims that Ruby is easier to learn than Objective C. After some research it seems that it’s not particularly easy to fully understand Ruby (even with a lot of programming experience in other languages than Objective C and Ruby). Some logical connections could be made at some point, but putting the newfound knowledge to the test to create an application did not go too well.

Below the online interface of Rhomobile is shown first. It is called RhoHub. The image on the next page will show how it looks.



```
# Startup page for your application
start_path = '/app'

# Path to the options page (in this case handled by javascript)
options_path = '/app/Settings'

# Location of bundle url (i.e. from rhohub.com)
rhobundle_zip_url = 'http://iconcept.rhohub.com/apps/1389/bundle/2802d51d-dcc6-4701-bc45-dc9a82c7d980'

# Optional password to access bundle (usually not required)
rhobundle_zip_pwd = nil

# Rhodes runtime properties
MinSeverity  = 0
LogToOutput = 1
LogCategories = *
ExcludeLogCategories =
KeepTrackOfLastVisitedPage = 0
LastVisitedPage = ''

# Sync server url.  This must be the full path to your sources.  For example: http://rhomobile.rhohub.com/apps/SugarCRM/sources/
syncserver = 'http://iconcept.rhohub.com/apps/RSSReader/sources/'
```

A screenshot of RhoHub, the online interface developing environment of Rhomobile

The above screenshot shows the editor that allows to write or make changes to the application. On the left a file structure is shown and on the right is where the source code will be inputted using HTML, CSS and Ruby. This is basically all there is to it. In order to start writing the application digging deeper into Ruby was needed and how to combine it with RhoHub.

To study Rhomobile (and RhoHub) there were some tutorials from Rhomobile, but the documentation on it is very little. It is possible to write a simple application where some details can be entered (for example the name and price of a shop item), which would then be saved locally on the smart phone. Every detail entered would be neatly listed vertically in alphabetical order. Now even though that sounds good and was not too hard to accomplish, that is about as far as one could get with all the documentation that Rhomobile has.

Another thing about Rhomobile is that it seems hard to alter the look and feel of an application. Most components have a very standard and basic look (in all operating systems) and it takes some effort to changes these, let alone edit the graphical user interface to your satisfaction. Aligning some components horizontally and others vertically is also a challenging task. In other words, one really needs to be up to speed with Ruby (and RubyOnRails) in order to work with Rhomobile.

Luckily there is an alternative, so let’s check out Steape.

## Steape

The Steape website looks more professional, but promises to do the same as Rhomobile. That is create an application one time and then have the option to deploy them on the five major operating systems. The IENS application mentioned earlier (in a previous chapter) is already available for three of them. So let's see if there's any truth in that and start building the application with the help of the online tool that Steape provides.

There is a technical whitepaper about the Steape platform on their own website if one wishes to learn more about it in detail[13].

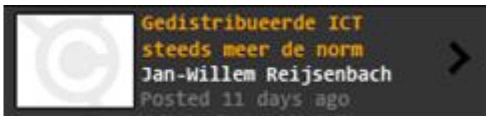
Let's sum up a few things, to give a quick impression of the Steape dashboard:

- Drag and drop functionality
  Simply add elements to the screen by dragging and dropping a component on it. So if the set up of the application is already known, all components can be placed very quickly.
- Editing components
  After a component is placed, it can be selected and its properties viewed. It's also possible to edit them appropriately. For example: if an anchor is placed and it needs to be a textual link instead of an image, it can be told which destination it should link to and more. It is an easy way of managing the application and it does not require to learn a complicated programming language.
- Visual placement of components
  It takes some time getting used to the visual interface of the Steape dashboard. Even when figured out it is easy to get confused. Especially so when components have to be aligned horizontally within other components and such. The more components there are, the more unorganized the screen will look. Unlike the Flash IDE  there is no preview of the application and it's not possible to move components just by dragging it with the mouse.

- Under development!
  The Steape dashboard is undergoing a lot of change. Almost every week something new is added or the graphical user interface is enhanced.

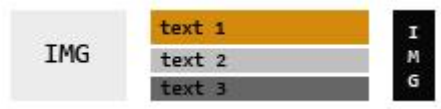
Now that the dashboard looks a bit familiar, let's begin with the application. To help understand how to place the components, let's use an example from one of the screens shown on page 24.

[13] Technical whitepaper | Steape

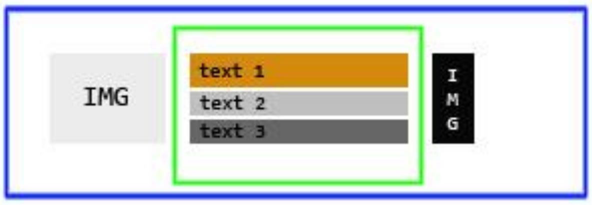


This is part of the screen, but is enough to show how to place components using the horizontal layout and vertical layout. This is one of the most important things to keep in mind.

To create something equal to this image, seven components are needed. One vertical layout component, one horizontal layout component, two image components and three text field components. The reasoning behind it is shown below. The next set of images should make it more clear.



The image can be broken up in five parts (not including the background color). Two images and three different text fields. The three text fields are aligned vertically. The whole is aligned horizontally though. The following image shows the containers.

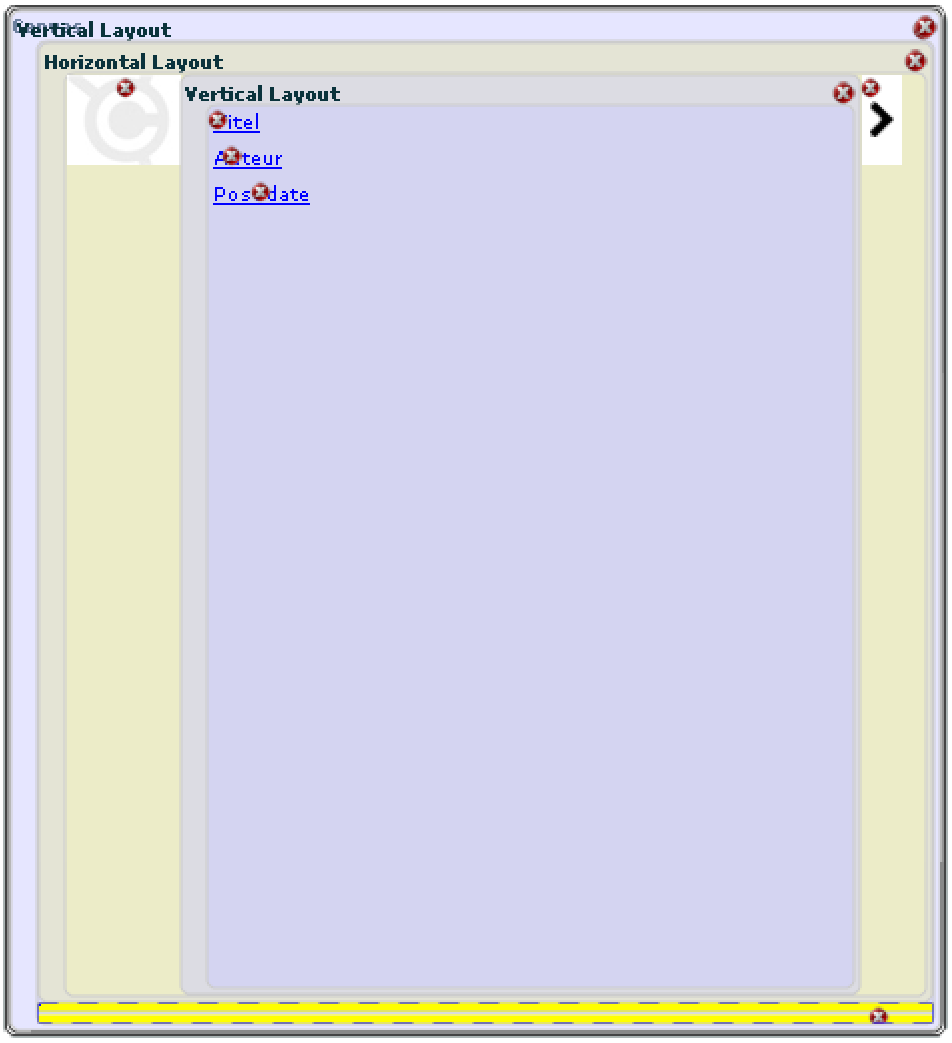


The blue container is the horizontal layout component. Within this component there first is an image, then a vertical layout component and another image. The vertical layout (the green container) has three text fields. So it is possible to layout components within each other (infinitely). This is why it can get confusing when there are a lot of components in the screen.

So in this example the order in which we place the components is:

1. Horizontal layout
2. Image
3. Vertical layout
4. Text field
5. Text field
6. Text field
7. Image

The example represented just one weblog entry that was viewable. If several of these entries needed to be listed, all of them could simply be put inside a parent container, which should be a vertical layout component. The example just described would look like this in the dashboard environment (see image below).



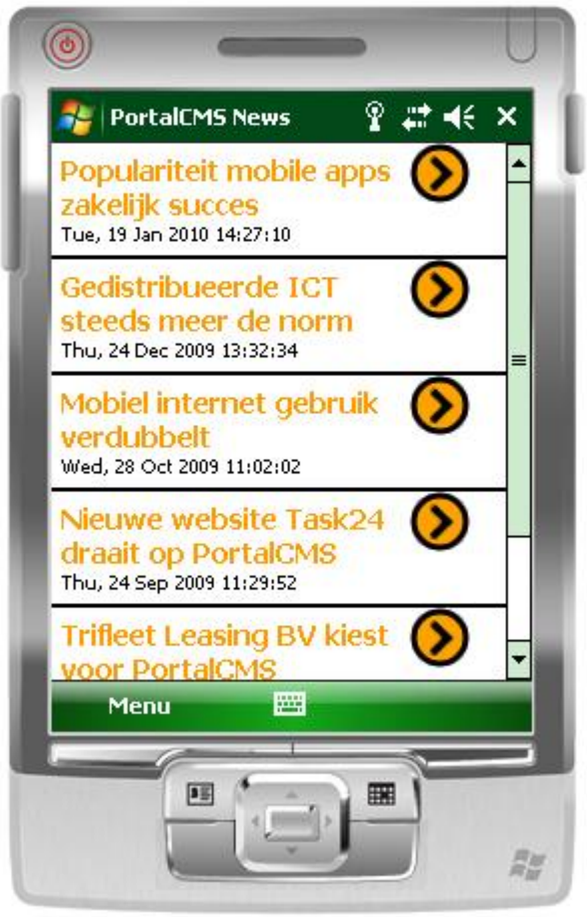
Now that it's clear how to set up the application visually, it's time to start adding some functionality to it. The news items that need to be listed will come from the following source:

http://portalcms.nl/nieuws/rss/ (this is an RSS feed from one of i-Concept's products)

In order to load the data from the RSS feed, the repeater component will be needed. The name is fairly self explanatory: it repeats one action. In our case, the title of the first post is requested and the screen "writes" the graphics to the screen with the proper title, then it repeats the same for the second post, and so on. This way our screen doesn't have to be filled with components in order to visualize a long list of items. The repeater component can do that.

A few problems occurred when setting up the graphical user interface for the application. It seemed surprisingly hard to 'tweak' the looks of an application, despite the drag and drop interface of the dashboard. For example outlining an item (like vertical centering) does not behave as wanted. This is something Steape will definitely have to improve in the future, because if they cannot give the developer a solid way of defining their design, they might lose potential customers for that particular reason.

The result of all news items being shown on the mobile screen is shown in the image below. Note that the goal was to have the orange arrow images outlined to the right of the screen and vertically centered. Instead, the images are clinging to the top of each row, they are not centered. They are also not outlined to the right of the screen, there is a white space there still. After a lot of editing, it seemed impossible to position all items the way they should.


PortalCMS News

Populariteit mobile apps zakelijk succes
Tue, 19 Jan 2010 14:27:10

Gedistribueerde ICT steeds meer de norm
Thu, 24 Dec 2009 13:32:34

Mobiel internet gebruik verdubbelt
Wed, 28 Oct 2009 11:02:02

Nieuwe website Task24 draait op PortalCMS
Thu, 24 Sep 2009 11:29:52

Trifleet Leasing BV kiest voor PortalCMS

One of Steapes current problems is that applications generated with their dashboard cannot store data on the smart phone. That means that there always has to be an internet connection available in order for our news reader app to function properly. Without the help of online webservices this application wouldn't have worked at all. Basically a website had to be created, that recognises a command that is sent from the application, processes that command by selecting the right newspost(s) and returning only that data to the application directly, where normally all data (from all newsposts) would have been returned. Usually the task of selection is performed by the application itself.

What did not seem to work was the ability of writing a new web log entry with this application. It requires too much data that has to be locally stored, most importantly a user name and password that is required to post something. There is a form component that could work, but that would only work without a login process.

## Results

After more research on Rhomobile documentation it seems very hard to understand the basics of the Rhomobile online IDE, even for experienced programmers. It would be even harder for a developer that has very little experience with other programming languages. Even if you succeed building an application, there is very little alteration you can make graphic wise. Or perhaps there is, but there is no documentation on it right  now. Hopefully, the more Rhomobile becomes widely known, the more documentation we might see appear on the internet. As for now, for a company that is not familiar with mobile development, Rhomobile is not a great option.

Steape has taken great steps towards 'write once, run anywhere'. It is really easy to understand their dashboard IDE and allows you to edit a lot of properties of components in a simple way. Steape seems to have released their service too early though, because it lacks correct documentation (which is being revised by another company at the time of writing) and the online dashboard is not without flaw. Especially the fact that data cannot be stored on the device the application is running on is a major disappointment. Furthermore it is very time consuming to get the application to look like you want it to, if you even manage to pull it off.

Even though we could not fully finish our application, we managed to get the majority of the functionality in there. A basic XML (which is the markup of an RSS-feed) reader is possible and written very quickly with the help of the Steape dashboard.

Since the case application didn't work out perfectly, another application is demonstrated to show that the 'write once, run anywhere' is indeed achieved. The company IENS has chosen to generate their application for three operating systems: Windows Mobile, Android and iPhone[14]. All three versions generate the same content and look only slightly different from each other.

[14] De nieuwe IENS applicatie nu ook gratis voor Android- en Windows Mobile-toestellen

## Conclusion

The following question is what was researched:

> **How can a company not already familiar with mobile development achieve 'write once, run anywhere' to create mobile standalone applications for smart phones?**

Currently existing and widely known options were reviewed, but the common problem is that any option will at most support three of the major platforms. This is the case for Adobe Flash Lite as it is for J2ME (these two are the most widely used options for creating mobile applications in the last few years).

New and upcoming technologies like Adobe AIR for mobile, Rhomobile and Steape are very promising. Adobe AIR however has not been released for mobile devices yet. Rhomobile has a very steep learning curve and lacks proper documentation in order to really create an application easily. Companies that are not already familiar with mobile development will not be able to finish a working application without it.

Steape currently seems to have the best cards on the table. The two most important reasons are:

- Create an application once and then generate versions for smart phones with different operating systems with only a few mouse-clicks.
- The online dashboard (with drag and drop functionality) is easy to learn and even gives people with no experience in mobile development the chance to write applications.

Let's take a look at the research question again. It can now, with confidence, be said that Steape has the solution to the problem. Despite the flaws that it has when it comes to designing a application visually and the lack of data storage capabilities, it covers the aspects that were set out to research. Steape has achieved 'write once, run anywhere' and it allows companies that are not already familiar with mobile development to create their own standalone applications for smart phones, despite their probable lack of experience in that area.

## Recommendations

Based on the results of the research and the knowledge was gained in the area of mobile development, there are a few short-term suggestions that can be made.

If you are looking for an easy-to-use platform and just want to get started with mobile development, Steape is a good place to start. You can sign up for a demo account and try it out. Don't expect an amazing application though, since Steape has a lot of flaws that need to be polished and new functionalities still have to be added. Simply loading external data is no problem with Steape, so an application like a news reader should work out fine with the help of Steape.

If an application with an advanced graphical user interface needs to be created, the best choice is to write an application for a specific device or operating system. Steape does not have the tools to precisely place components or edit them accordingly. Better would be to use Adobe Flash Lite when developing for Windows Mobile, since Flash will show exactly (pixel-perfect) what the application will look like and has the ability to create vector-based animations.

If as many devices as possible need to be reached, it would be better not to use any of the discussed developing methods, but instead focus on creating web applications for mobile devices. Web applications, unlike standalone applications, do not need to be installed or be of a specific programming language. The mobile browser on a smart phone almost always supports web applications of many kinds.

If the intention is to create an application that handles a lot of data, Steape is not the solution. It cannot store data and send it to another screen unfortunately. Again, it is wise to just target a specific operating system and develop the application for that alone. Building an application in its native language has a lot of benefits and has been done countless times with great applications as result. The downside to it is that sufficient knowledge of the programming language for that operating system is required.

So the recommendation to people (and companies) is to think hard about what the final goal of the application is. Companies that want to create mobile games should not even look at Steape. But those that want to keep people informed of news about their company can write a newsfeed very quickly with Steape and then offer the application for multiple operating systems. It is all about the wishes of the companies and the limitations of the Steape platform.

Choose wisely.

## Summary

In the **introduction** it was found out that developing standalone applications is a trend. More and more companies are starting to develop their own apps, but for companies with little experience in mobile development it is hard to do so.

The research question is as follows:

**How can a company not already familiar with mobile development achieve 'write once, run anywhere' to create mobile standalone applications for smart phones?**

In **chapter 1** a clear picture of the properties of a smart phone was established. A smart phone runs on one of the five major operating systems, has access to the internet, has a touch screen, a QWERTY keyboard and allows the user to install new software (apps).

In **chapter 2** web applications with standalone and native applications were compared. Similarities and differences are listed and an explanation is given for the choice to develop standalone/native applications.

In **chapter 3** the native programming languages of each of the five major operating systems were looked at. Also some established and well known methods of developing mobile applications were reviewed. These include Adobe Flash Lite and Sun Microsystems Java Micro Edition. None of these are capable of 'write once, run anywhere', but some might expand their capabilities in the future.

In **chapter 4** some fairly unknown or upcoming technologies were researched in order to create applications which can be deployed on the five major operating systems. Two options really jump out, Rhomobile and Steape.

With our **case study** an attempt was made to build an application with Rhomobile and Steape. Rhomobile proved to be particularly difficult to a developer with no programming experience and it has very little documentation. Therefore Steape was tried next and it showed that an application can successfully be written and deployed to the five major operating systems. Some negative aspects were encountered too though.

The **conclusion** was made that while Steape does give a solution to the research question, it might not be the best option for every developer, because it does lack functionality. For example, data cannot be stored on the smart phone with an application written with the help of the Steape dashboard.

Finally, the **recommendation** was made to companies, to look at what kind of application they are trying to build and keep in mind the limitations that the Steape platform has. If they do that, they should be able to make an informed decision about what platform they can choose in order to write their applications.

# Sources of information

**Internet sources**

Global Handset Shipments Fall 4 Percent in Q3 2009, but Return to Growth Expected in Q4
http://www.businesswire.com/portal/site/home/permalink/?ndmViewId=news_view&newsId=20091029006638&newsLang=en
Consulted on: 03-11-2009.

App store downloads exceed 100 million per month
http://www.fiercemobilecontent.com/story/app-store-downloads-exceed-100-million-month/2009-11-18
Consulted on: 14-12-2009.

Developing Java-Based Mobile Games
http://www.developer.com/java/j2me/article.php/3502741
Consulted on: 17-12-2009.

Is Flash better than Java?
http://blogs.zdnet.com/Burnette/?p=286
Consulted on: 17-12-2009.

Vector graphics
http://en.wikipedia.org/wiki/Vector_graphics
Consulted on: 18-12-2009.

Bright 25 van 2009
http://www.bright.nl/bright-25-van-2009
Consulted on: 18-12-2009.

Silverlight for Mobile
http://silverlight.net/learn/mobile/
Consulted on: 16-12-2009.

Flash Lite Penetration Forecast 2008-2009
http://www.flashmobileblog.com/2008/11/08/flash-lite-penetration-2008-2009/
Consulted on: 16-12-2009.

Qt – A cross-platform applications and UI framework
http://qt.nokia.com/products
Consulted on: 23-12-2009.

De nieuwe IENS applicatie nu ook gratis voor Android- en Windows Mobile-toestellen
http://www.iens.nl/nieuws/2009/1/bestel-iens-mobiel.html
Consulted on: 30-12-2009.

Technical whitepaper | Steape
http://www.steape.com/steape/Steape%203.0%20-%20Technical%20White%20Paper.pdf
Consulted on: 12-10-2009.

# Attachment: IENS application

## About IENS

IENS is a company that reviews restaurants all around the Netherlands. Their website allows anyone to submit reviews and lets you browse through countless restaurants, based on location or rating. They recently released an application that has the same functionalities as their website. This application is available for the iPhone, Windows Mobile and Android devices. Why they chose to not release the Symbian and Blackberry versions is unknown.

## The application

Let's sum up the things can be done with the IENS application.

Search by…

- restaurant name.
- location.
- current location.
- kitchen.

As shown there are several ways to find restaurants. First of all a simple search by restaurant name can be done. This will usually result in the restaurant of your choice, unless there are several restaurants with the same name. If there are several results, restaurants with the highest rating will be listed first.

Search by location can also be done. If one lives in Amsterdam, a search for restaurants in that particular city can be made. A region or part of that city needs to be selected. After that a preferred kitchen can be selected. For example Chinese, fast food or Greek. No preference can also be selected and it will then show all restaurants in the selected area.

With the help of GPS, restaurants that are close by can be searched for as well. Just like any other search method, the restaurant with the highest rating is listed first.

The application does not stop there though, there is one more thing that can be done: submit a review. Once the correct restaurant is found, some text about the experience can be written and an one out of five ratings can be given: excellent, good, decent, average or bad.

This IENS application has been created with the Steape platform. They designed it and added functionality only once, then with a few clicks of the mouse they could generate three different versions of the application.

To download and view the different versions of the IENS application, you can visit their website: http://www.iens.nl/.